Sugarizer Knowledge Activity Pack

About Me

Name : Prakash Ujjwal
Email: prakashujjwal1010@gmail.com (Primary),

prakashu.co18@nsut.ac.in (Professional)
Github: https://github.com/prakashujjwal1010
Languages: Hindi (Native), English
medium: https://medium.com/@prakashujjwal1010

Location: I am located in New Delhi, India and my time zone is Indian Standard Time (UTC + 5:30). I am planning to work from 6:00 to 16:30 (UTC) but my timings are flexible. I'm very excited to work on this project and I can surely manage and adjust my time and be active when the mentors are available.

Degree : Bachelor of Engineering
Major : Computer Engineering
Institute : Netaji Subhas University of Technology, New Delhi (formerly NSIT)

What is your motivation to take part in Google Summer of Code?

GSOC is one of the most prestigious annual coding programmes which provides a very good opportunity to undertake projects in the midst of a community that leads to a lot of learning and growth.

Why did you choose Sugar Labs ?

Sugar Lab is a community that focuses on providing an open-source learning platform for children which uses technical platforms for a societal cause.

I too want to be a part of Sugarlabs which helps many children to learn something new in a fun way.

The idea of helping so many children by the programme i will code and the satisfaction it will give will be magnificent.

Why do you want to work on this particular project ?

The problem statement of the project is very interesting and the project is using technologies which aligns with my skills and which I have worked upon before.

I have developed a few sugarizer activities before also which makes it very intriguing for me to choose and work on it.

What are your expectations from us during and after successful completion of the program?

I expect to be mentored by the community during the GSoC period so that I can learn and contribute at the same time which will, in turn, help me to add value to the community and help others. Plus I expect the active communication between me and organization and mentors.

Skills : Js, Python, C++/C , Reactjs/ Redux, Vuejs, Nodejs, Ethereum Dev, Truffle, HTML, CSS, Docker, Flatpak.

Projects

I have experience in web development and decentralized apps development.

1) **Split-Party:** DAPP built upon ethereum blockchain to plan, manage, share, track your party expenses in a trustless way. It is developed using Reactjs, truffle. The main idea was to use ethereum in some fun and unique way.
   https://github.com/prakashujjwal1010/Split-Party

2) **Stubble-Trouble :** it is a group project. Here, In India , Stubble burning is a very big problem (which is one of the reasons for pollution and smog in new delhi around november). We are aiming to provide a marketplace and an online self-sustainable business model which would connect farmers to potential buyers of stubble. In this way the stubble burning could be stopped and could be used in an efficient way.
   We got an offer from our Institute's E-Cell for funding.
   It is not yet completed. We have plans to continue working on it after a few months(after GSOC).
   https://github.com/prakashujjwal1010/SIH2020-1

3) **Image-Uploader:** It is an implementation of an image uploader which sends an encrypted image to the server and then the server decrypts it and stores it. https://github.com/prakashujjwal1010/image-uploader

Open Source Contribution

I have some experience in open source.
I have been contributing to the open source community for more than 1 year which includes contribution in sugarlabs, scorelab etc.

Sugarlabs Contributions

I started contributing to sugarlabs and started exploring its various projects in september 2019.I started to give attention to all the discussion that used to take place in the sugar-devel list.
During this journey I learnt many new things (e.g. flatpak, localization)

Pull requests:

- porting Falabracman Activity to sugarizer (see live)
- added set-time and set-time-game mode in clock activity (see live)
- added changing generation speed feature in Game of Life Activity (see live)
- added changing grid size feature in Game of Life Activity (see live)
- added a feature to display trails of cell's deaths (see live)
- chess Activity for sugarizer as GSOC task
- Packaging of sugar activities to Flatpak
    - packaged Falabracman activity to Flatpak (or here)
    - packaged Napier activity to Flatpak(or here)
    - https://github.com/tchx84/flathub/pulls?q=is%3Apr+author%3Aprakashujjwal1010+
- fixing disappearance of clock in set time mode

- fixing responsiveness for falabracman sugarizer activity
- fixing end game screen issue after restart
- enabling touch events in XOEditor sugarizer activity
- added license tag
- fixing clear and save icon in XOEditor

Issues:

- touch events not enabled in XOEditor
- Stop Watch Activity is not responsive and has text overflowing problem.
- Bug While Resizing game screen after resizing setting screen in Falabracman activity
- Too random Generation of position for obstacles
- Activity needs license metadata for AppStream file
- reappearance of tutorial on resizing window after clicking next button on last step of tutorial

Others:

- sugarizer vue activity template

Contributions to Other Organizations

1. **Scorelab:**
    - **Pull requests in Senz**

2. **Publiclab:**
    - **https://github.com/publiclab/image-sequencer/issues/1241**
3. **NSITulatorWeb:**
    - **https://github.com/NSITulator/NSITulatorWeb/pull/6**

ABOUT THE PROJECT

The name of the project is Sugarizer Game Activity Pack which consists of two activities - Mind Math activity and Tangram activity.
These two activities are requested by Sugarizer deployment in Saint-Ouen.

Mind Math Activity:

The Mind Math activity will be a game to practice mathematics in a different way. The student is given five random numbers and should use the four basic arithmetic operations (+, -, ×, ÷) to build an operation that will result in the given output.

The activity will have following detailed features:

- The activity or game will give 5 random numbers(one between 1-4, one between 1-6, one between 1-8, one between 1-12 and one between 1-20) , a target number, 4 operators to the users and 4 slots to find the chosen or target number.
- The game will provide a way to the users to select their choice of numbers and operators.
- The game will show the operations performed so far by the user during a game.
- There will be two types of difficulty level in the game.
    - an easy level with a target number between 10 and 69
    - a medium level with a target number between 0 and 99
- The difficulty of a game can be changed by making some operators compulsory.
- The users will get a score also at the end. The score will depend on
    - Number of slots used (more is better)
    - Bonus when all four arithmetic operations are used
    - Bonus depending on the time spent to solve the challenge
- The game will have two modes: single player and multiplayers.
- In single player,
    - The game will have an undo feature.
    - The game will have a restart feature.
    - There will be a feature to change the difficulty of the game.

  - There will be a feature also to make some operators compulsory.
  - There will be a solver.
  - There will be a feature to get hints during the game from which the users can take help.
  - There will be a feature for enabling and disabling the timer also.
  - When the timer is disabled,
    - There will be no timer
    - The screen will show the **time spent** by the user for a question and it will affect the score also.
    - After every question, **time spent** will get initialized to zero.
    - The length of question set of the game will be 1 in this case
    - The new game with a new target number and 5 numbers will come after the user won or completed the previous question.
    - After every question completion a game-end modal or popup will appear showing the best result and score.
  - When the timer is enabled,
    - There will be a timer
    - The screen will show the **time left** by the user for a question and the score will be dependent on the time spent by the user so indirectly depends on the time left.
    - **Timer** will be initialized only after the timer ends.
    - The length of question set of the game will be 1 in this case
    - The working of the activity will be like a **sprint math activity** in this case. The user will be given a time in which he can score as much as possible by completing questions by questions. The new game with a new target number and 5 numbers will come after the user won or completed the previous question. The end score will be the sum of all the scores of the questions completed in one stretch or game. In other words, users have to make as many points as possible in a limited time where you will get question by question.
    - After every question completion a milestone modal or popup will appear showing the best result.
    - After the timer ends, the game-end or a timer-end modal or popup will appear showing the total score.
- In multiplayers mode,
  - There will be no feature for undo, restart, change difficulty, change compulsory operators, enable/disable timer. All of them will be disabled in multiplayers mode.
  - There will be no solver.

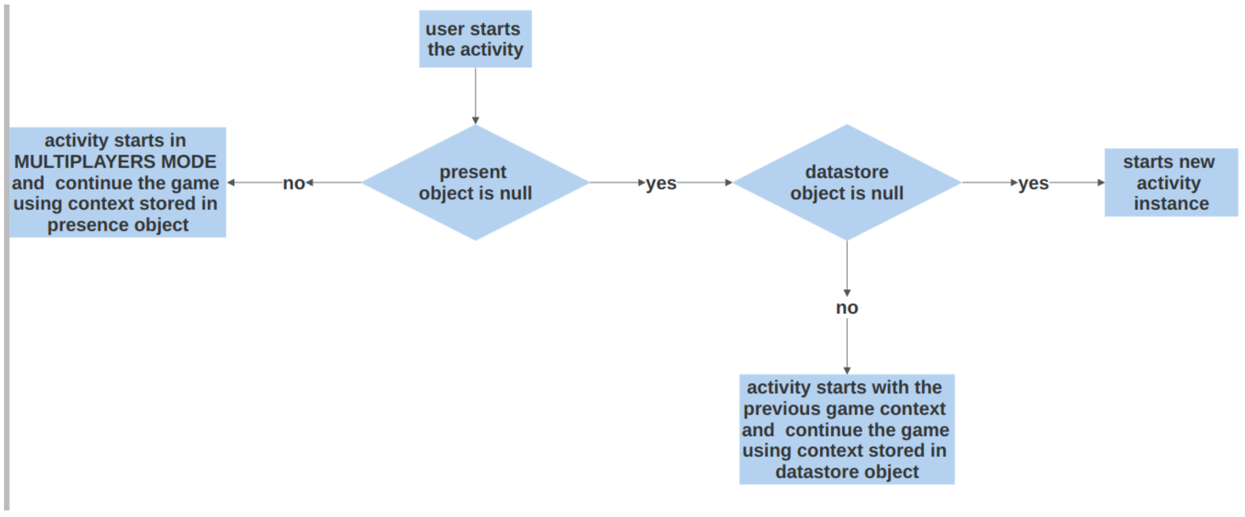
  - There will be a timer as in multiplayers mode the timer is always enabled.
  - users have to make as many points as possible in a limited time where they will get question by question.
  - Users will have a leaderboard also in which they can see the end scores of every user connected via presence.
  - After every question completion a milestone modal or popup will appear showing the best result.
  - After the timer ends, game ends, a timer-end modal or popup will appear showing the total score.
  -
- Activity should be localized, responsive, should have a tutorial, have a fullscreen button.
- The context should be stored using a journal.

Note:

- Negative numbers and fractions are not allowed. For example,while performing an operation, you can’t subtract a 7 from a 2, as that would give -5, and likewise you can’t divide 2 by 7 as that gives a fraction.
- When the timer is **enabled,** The game will end when the timer ends **but** the question will end when the user finds the target number and Each game has a number of questions or calculations.
- When the timer is **disabled,** The game will end when the user finds the target number and Each game has only one question or calculation.

Implementation:

Basic user flow diagram for starting the activity:


user starts the activity
present object is null
no
activity starts in MULTIPLAYERS MODE and continue the game using context stored in presence object
yes
datastore object is null
yes
starts new activity instance
no
activity starts with the previous game context and continue the game using context stored in datastore object

- **Generating question set**

  For generating a question for mind math activity, we have two options:

  - **Generating all the 5 numbers and a target randomly**

    This will have scenarios where there is possibly that there is no possible Way such that we can find the target number using given 5 numbers.So we should avoid this method.

  - **Generating the 5 numbers randomly and then generating the target number using these 5 numbers.**

    Here, we will always have a set of 5 numbers for which there is a solution or a way to find the target number.

    Steps:

    1. First we will generate 5 random numbers within a specified range i.e first number between 1 to 4, second 1 to 8 etc.
    2. Find the random ***valid postfix expression** (reverse polish notation) using these numbers and four operators.
    3. Calculate the result of the expression and store it as the target number for the question.
    4. For the target number generated or the question, the postfix expression is actually the **#best solution**.So we can store it with the question as a best solution also.

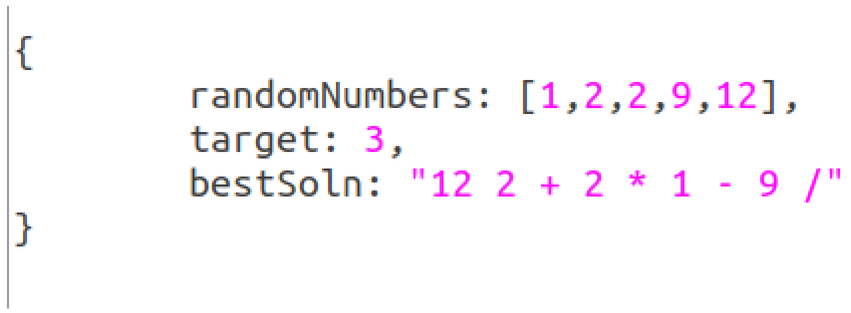
    ***valid postfix expression**: in the context of our game, valid postfix expression is which when evaluated no any negative or no fraction number appears. Plus it must have two numbers at the start and one operator at the end. Also, The target number generated via the expression should be according to the **difficulty level.**

    #best solution: the valid postfix expression from which the target number is generated, is a best solution because all the four numbers and all the four operators are used .

(note that there can be more than one solution for a question and more than one solution can be best, it's just that we have chose this as a best soln)

Example:

1. **1,2,2,9,12** are the numbers generated randomly within a specified range for each number.
2. A random valid postfix expression for these numbers with four operators is **12 2 + 2 * 1 - 9 /**
3. Evaluate the expression we will get **3** which is our target number.
4. So our question object would be :



```
{
            randomNumbers: [1,2,2,9,12],
            target: 3,
            bestSoln: "12 2 + 2 * 1 - 9 /"
}
```

Now, why are we generating a question set beforehand? Why not generate a question for a game on the fly?

Because, when the user will share the activity, all the users will get the same question set as the host.

- **Generating tips during the game**

Users will get the tips during the game.

The tips will show the next possible operation for a next slot or in other words, possible two numbers and an operator for the next slot.

In other words, we have to find all the solutions which are starting from the operations performed or slots filled so far by the user.

To find this, we will first convert the slots filled to the valid postfix expression.

then we will generate the remaining valid postfix expression for which the user can get the target number.

Illustration:

If the random numbers are **1,2,4,9,11** and target number is **4.**

The first slot is filled by user: **1 + 2 = 3**

The second slot is filled by user: **3 * 9 = 27**

1. The slots filled will be converted to valid postfix expression

   1 2 + 9 *

2. Find the remaining part of the expression using the remaining numbers (**4,11,27**) and the remaining number of operators (here any 2 from **+,/,*-**) that will lead to the target.

   For finding the above we can generate permutations of valid postfix expressions using numbers **4,11,27 and any 2 operators from +,-,/*.**

   11 27 + 4 - , 27 11 - 4 +,......., 27 11 - 4 /, …. …… etc.

   We will select anyone from the ones which produce the target number when evaluated.

   So we choose **27 11 - 4**

   We will call this step solution generation[$$] step.

3. We will express this expression to slots.

   27 - 11 = 16

   16 / 4 = 4

4. We will show only the one slot to user as a tip or help i.e

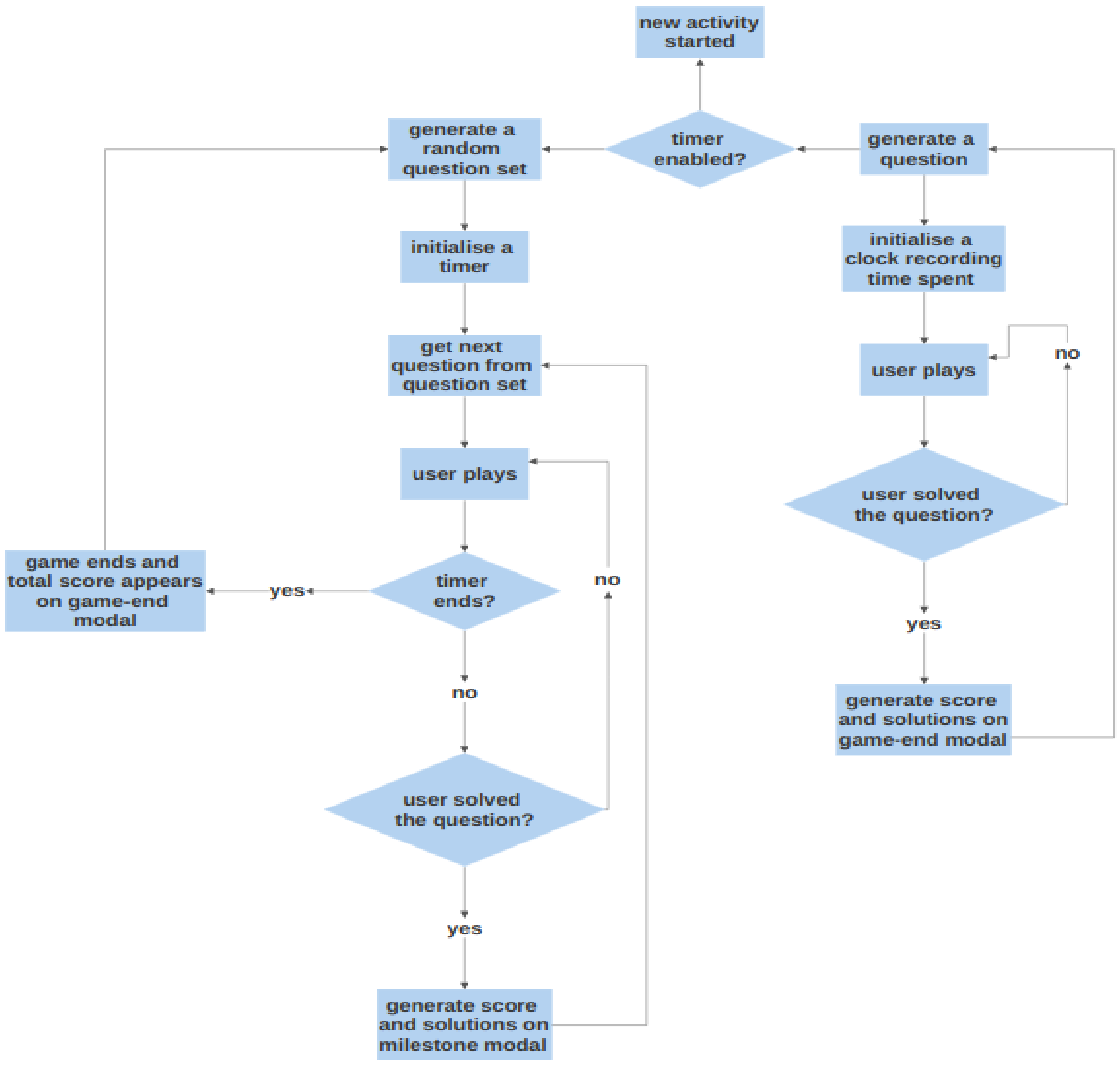
   27 - 11 = 16

[$$]**solutions generation:** we can take inspiration from following open source libraries to implement this step :

- Cntdn
- Hello-countdown-numbers-game
- Numbers

- **Playing the game**

  For single player mode when the user starts the new activity (**both datastore and presence objects are null**):


new activity started
timer enabled?
generate a random question set
initialise a timer
get next question from question set
user plays
timer ends?
yes
game ends and total score appears on game-end modal
no
user solved the question?
no
yes
generate score and solutions on milestone modal
generate a question
initialise a clock recording time spent
user plays
no
user solved the question?
yes
generate score and solutions on game-end modal

During the starting the game, we will fetch the question from the question set we have generated before or fetched from the journal or generate a question. Then we will store the 5 random numbers from the question in an array called **inputNumbers[].**

While playing the game to fill a slot , the user will choose two numbers from **inputNumbers[]** and an operator to fill a slot and then these two numbers will get deleted from the **inputNumbers[]** and the result of a slot will be pushed to the **inputNumbers[]**.

While filling the slot, users should take care of game rules i.e. no any slot’s result should be negative or fraction.

Users should use the **compulsory operator** at least once.

Users will have four slots(obviously).

- **Enabling / disabling timer**

  If the user enables or disables the timer, the new game starts automatically.

  Multiplayer mode has always timer enabled.

- **Calculating score:**

  Users will see their score after each game and a total score after the timer ends.

  The score will depend on:
  - Number of slots used (more is better)
  - Bonus when all four arithmetic operations are used
  - Bonus depending on the time spent to solve the challenge

- **Showing best result at the end:**

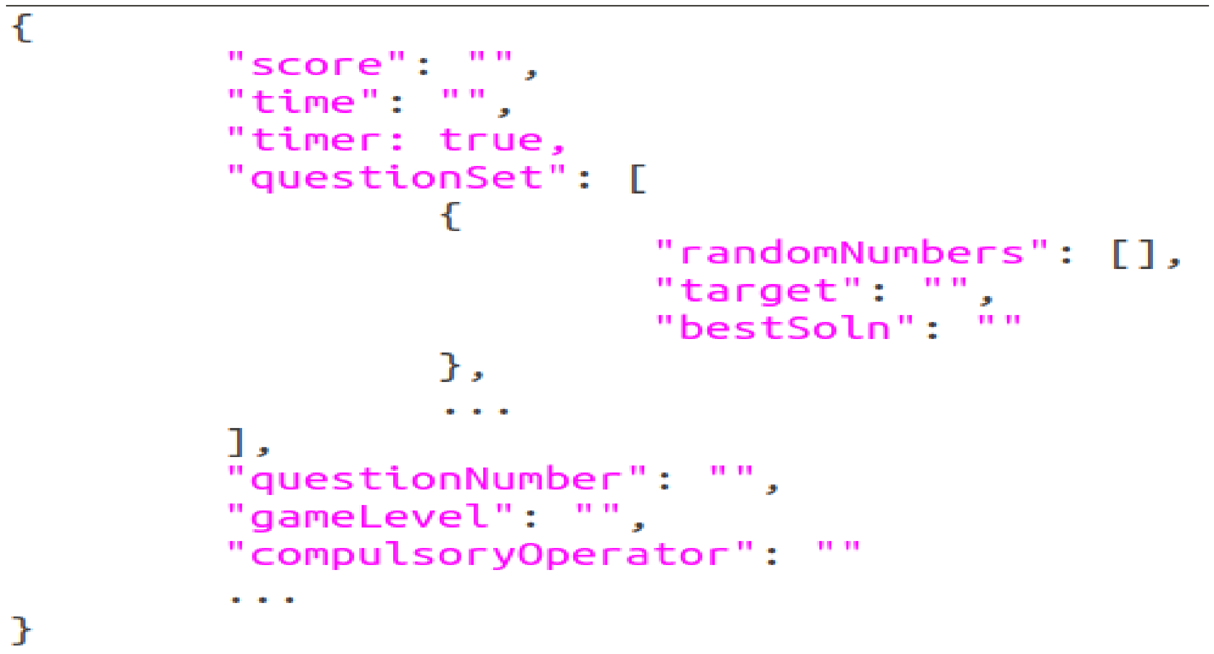
  In our question object, which we fetched from our question set, we also have the bestSoln field.

The field will contain the best solution in terms of postfix expression. To show it to users, we will express it in terms of slots.

Storage activity instance:

We will use a journal for storing the activity instance.

The datastore object will look like below:



```
{
        "score": "",
        "time": "",
        "timer: true,
        "questionSet": [
                {
                        "randomNumbers": [],
                        "target": "",
                        "bestSoln": ""
                },
                ...
        ],
        "questionNumber": "",
        "gameLevel": "",
        "compulsoryOperator": ""
        ...
}
```

(The datastore will have some more fields also.)

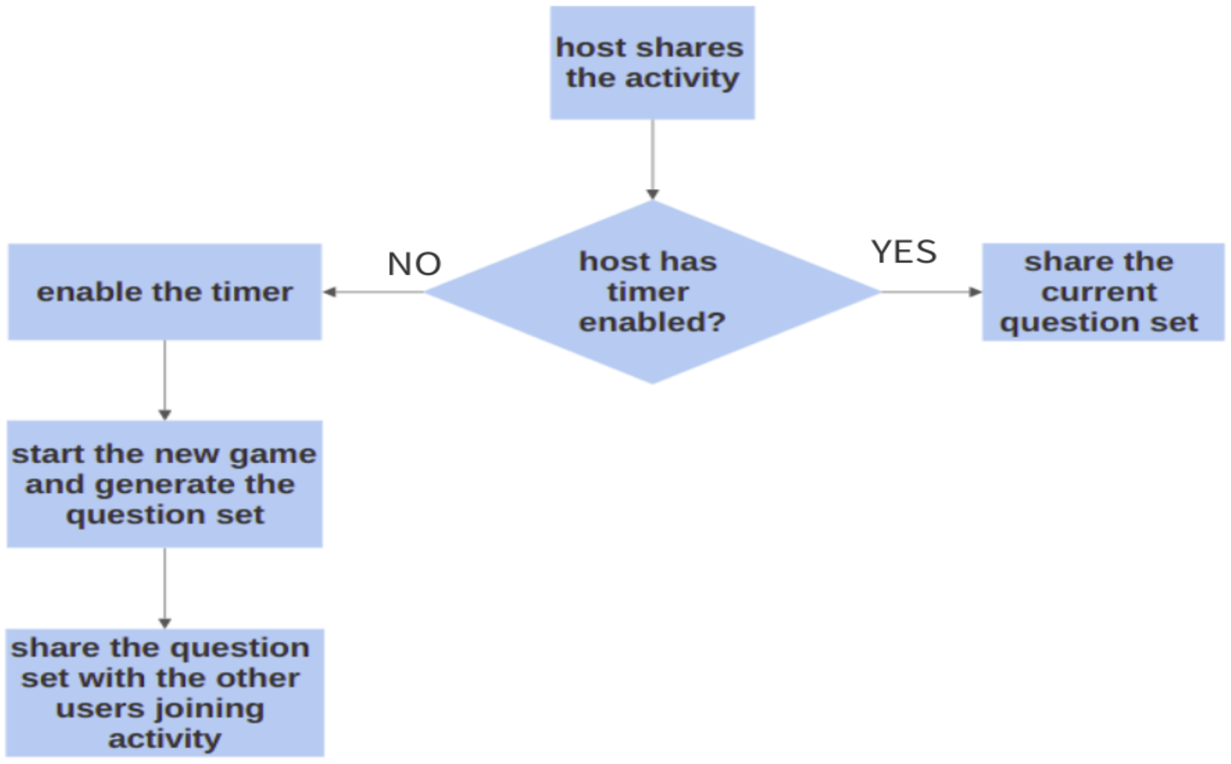
Shared activity or multiplayers mode:

We will use the presence here.
Once the host shares the activity, then if the host's timer is enabled then the host will share the current question set with users otherwise it will enable the timer then generate the new question set (i.e starts the new game) and then share the newly generated question set.
then users joining the activity via presence will get a question set and other info like difficulty level, compulsory operator etc.
Once the users end the games i.e. timer ends, he will get to see leaderboard.
And the leaderboard will get updated in real-time.

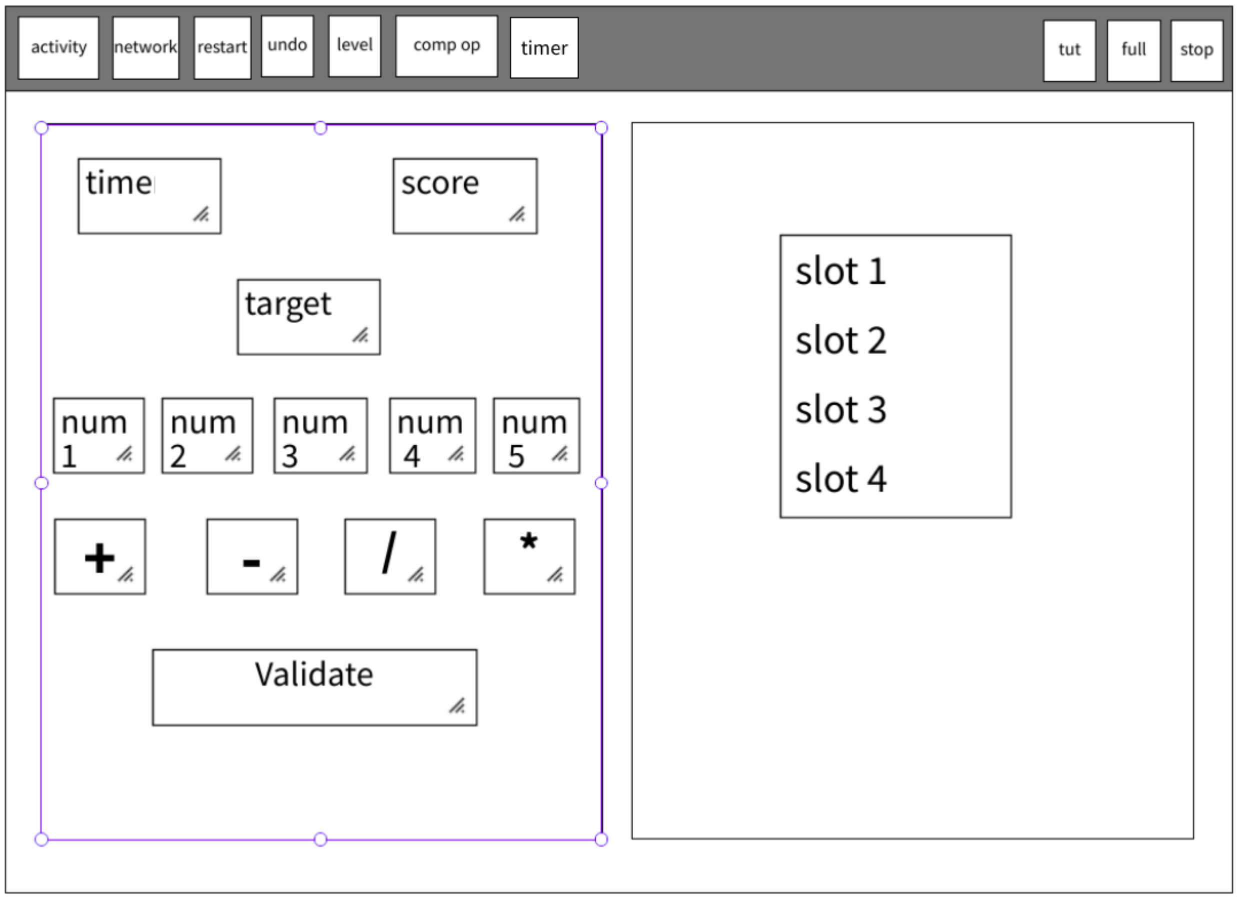
Presence actions :

- **Init:** for initializing the activity with question set,etc.
- **updateLeaderboard:** for updating leaderboard in real time.

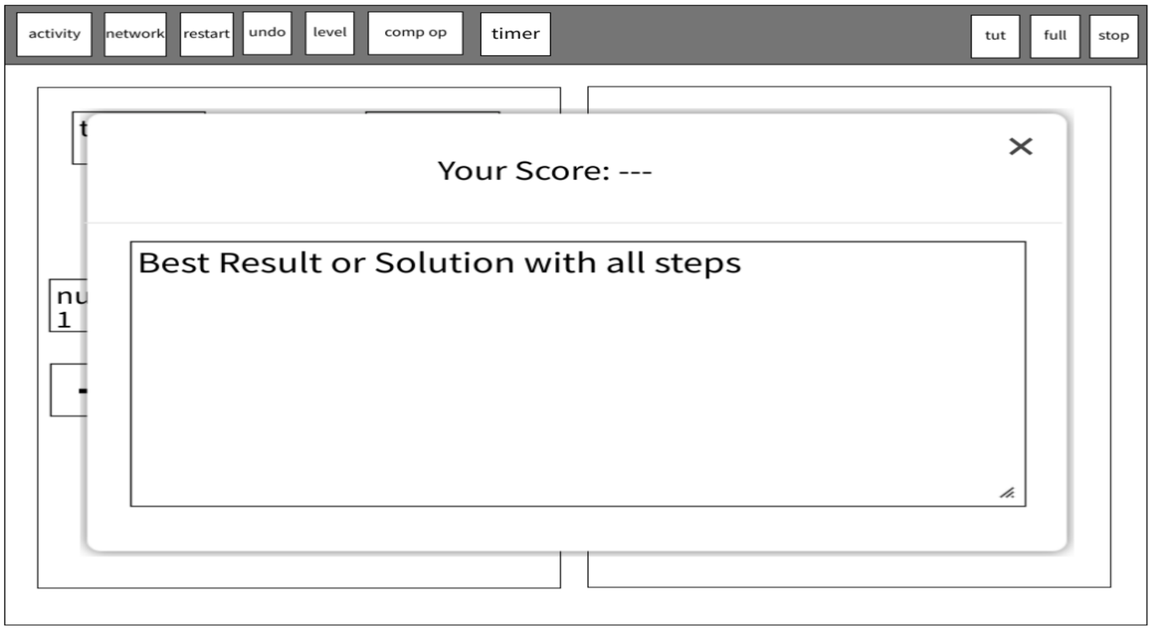
Host's Workflow for sharing activity


host shares the activity
host has timer enabled?
NO
YES
enable the timer
share the current question set
start the new game and generate the question set
share the question set with the other users joining activity

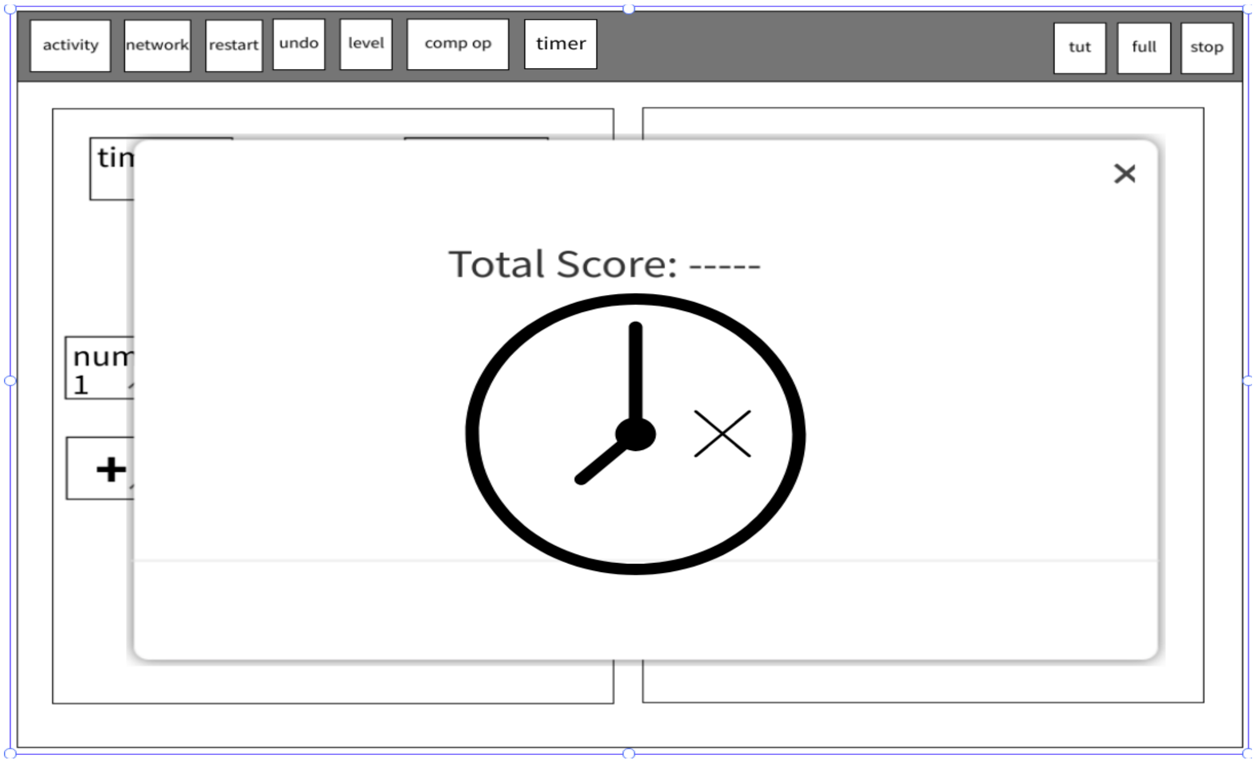
Basic Wireframe for ui:



- In single player mode, the screen will have two sections one which will show the numbers, operators, timer, score etc and another which will show the slots.
- If timer is enabled
    - “Time” will show the time left
- If Timer is disabled
    - “Time” will show the time spent so far for the question
- “Score” will show the total score so far.
- User will have a validate button which he will click when he wants to submit his answer for the calculation or question. The “validate” button will be enabled or visible when the target is generated during the operation or in slots.
- Once the user completes the question, a pop up will appear showing its score for the question and the best solution for the same with all steps(see below).



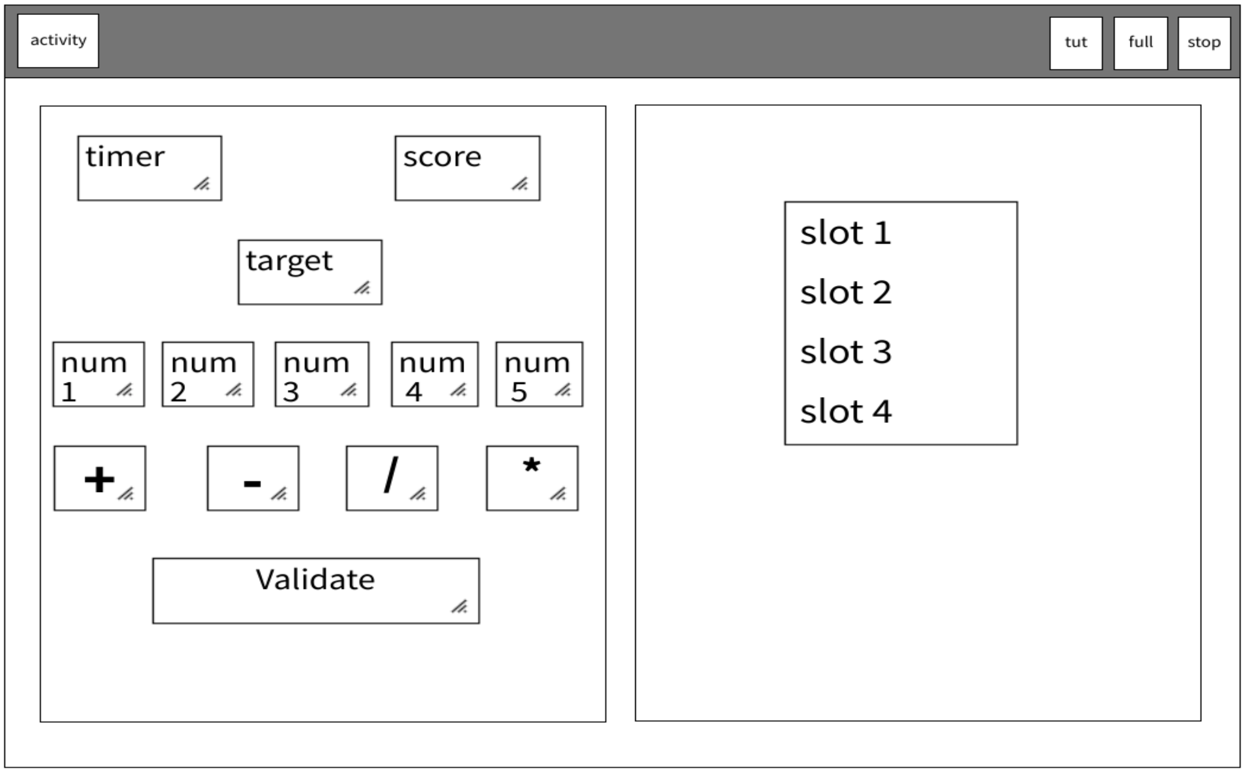
- If **timer is enabled**, then after the game completion or timer ends, the pop up will appear indicating that the time has expired and showing your total score.



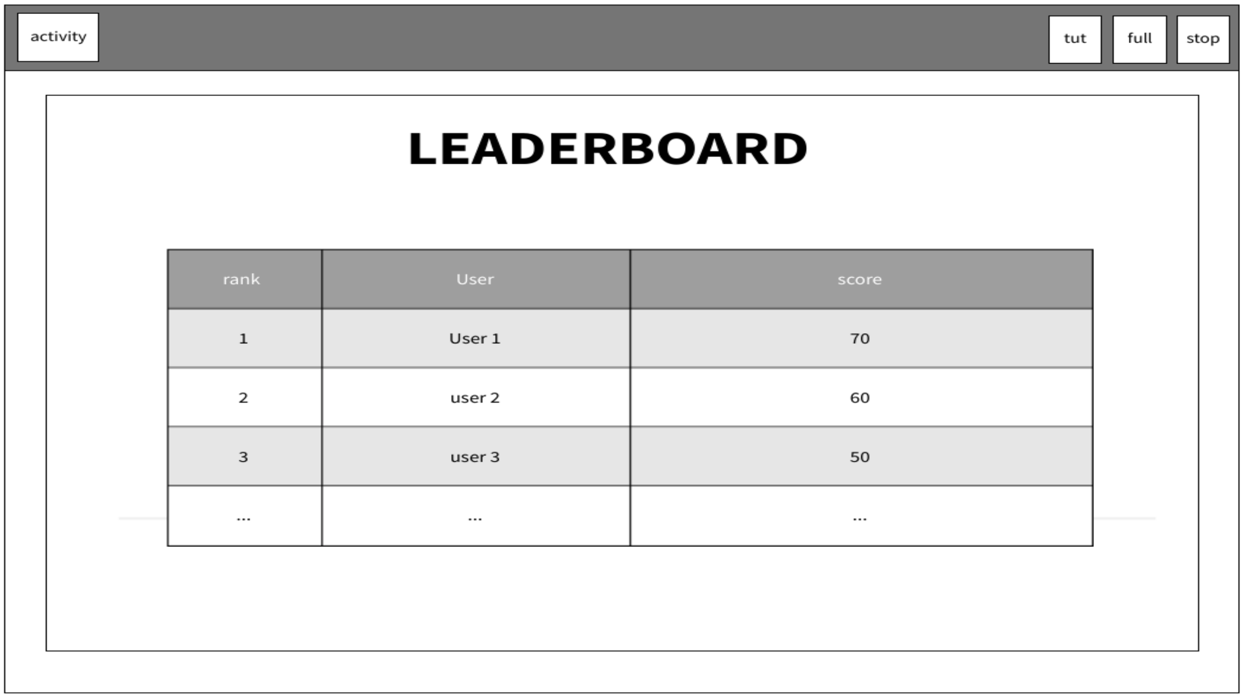
Toolbar in single player mode:

- **Activity:** activity palette
- **Network:** to share activity
- **Restart:** to restart the game with a new question set.
- **Undo:** to undo the step or operation performed.
- **Level:** to change the difficulty level.
- **Comp op:** to make an operator compulsory.
- **Timer:** to enable/disable timer.
- **Tut:** for tutorial.
- **Full:** for fullscreen/ unfullscreen.
- **Stop:** for stopping the activity or exit.

Multiplayer mode:



- In multiplayer mode, users will have almost the same ui as that of single player mode except the toolbar.
- After the game completion or timer ends, the leaderboard screen will appear which will show the ranks of users and their points in real time.



rank	User	score
1	User 1	70
2	user 2	60
3	user 3	50
...	...	...

Toolbar in multiplayer mode:

- **Activity:** activity palette
- **Tut:** for tutorial.
- **Full:** for fullscreen/ unfullscreen.
- **Stop:** for stopping the activity or exit.

Aim is to build ui as simple as possible.
For framework for ui: we will use vuejs
For building responsive design: we will use bootstrap.

Tangram Activity:

The Tangram activity will be an activity to play the traditional Tangram game.

The activity will have following detailed features:

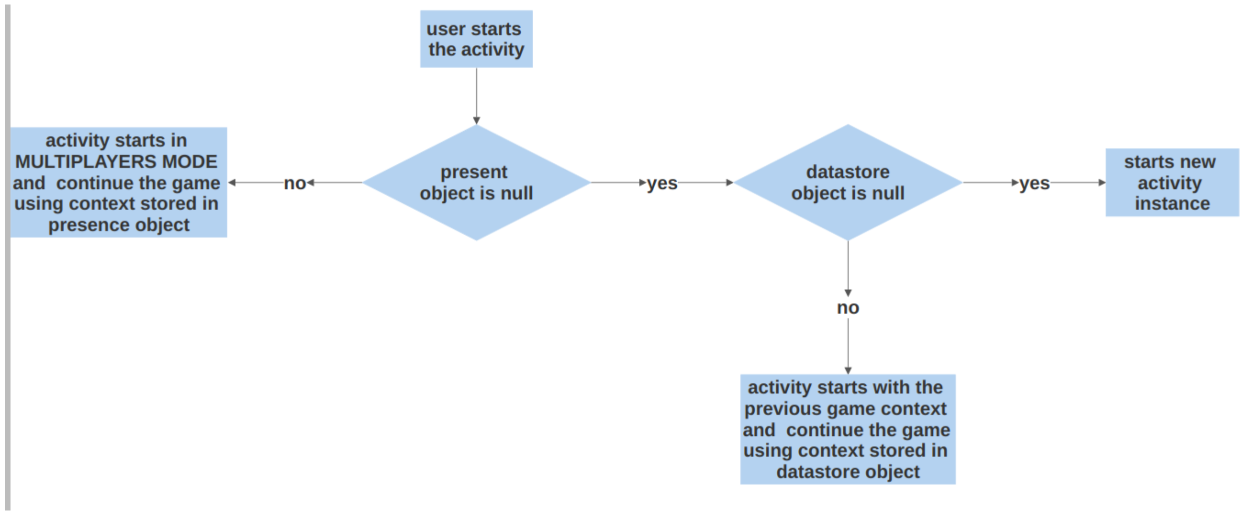
- The Tangram activity will present a set of Tangram pieces to the user, a specific shape and a canvas.
- the user should form a specific shape using these pieces on the canvas.
- There will be two types of difficulty level in the game.
  - an easy level where the user knows where each piece should be set on the shape and just have to move/rotate the right piece to the right place
  - a medium level where the user should guess where each piece should be set and move/rotate it to the right place
- The difficulty level could also depend on the complexity of the shape.
- The game will have two modes: single player and multiplayers.
- In single player,
  - We will have a feature to restart the game.
  - We will have a feature to change the difficulty level of the game.
  - We will have a feature to change the target shape level or complexity for the game.
  - The user will get a set of random target shapes of selected shape level.
  - There will be a feature for enabling and disabling the timer also.
  - When the timer is disabled,
    - There will be no timer
    - The screen will show the **time spent** by the user for a question and it will affect the score also.
    - After every question, **time spent** will get initialized to zero.
    - The length of shapes set of the game will be 1 in this case
    - The new game with a new target shape will come after the user won or completed the previous question.
    - After every puzzle or shape formed, a modal or popup will appear indicating the user has formed the target shape and his score.
  - When the timer is enabled,
    - There will be a timer
    - The screen will show the **time left** by the user for a puzzle and the score will be dependent on the time spent by the user so indirectly depends on the time left.
    - **Timer** will be initialized only after the timer ends.

        - The length of question set of the game will be 1 in this case
        - Users have to make as many points as possible in a limited time where once the user forms the target shape he will get the next target shape.The end or total score will be the sum of all the scores of the puzzles completed in one stretch or game.
        - After every puzzle or shape formed, a milestone modal or popup will appear indicating the user has formed the target shape.
        - After the timer ends, a timer-end modal or popup will appear showing the total score.
- In multiplayers mode,
    - There will be no feature for restart, changing difficulty, changing shape level. All of them will be disabled in multiplayers mode.
    - There will be a timer as in multiplayers mode the timer is always enabled.
    - All the users will get the same set of shapes as the host.
    - Users have to make as many points as possible in a limited time where once the user forms the target shape he will get the next target shape.
    - After every puzzle or shape formed, a milestone modal or popup will appear indicating the user has formed the target shape.
    - Users will have a leaderboard also in which they can see the end scores of every user connected via presence.
    - After the timer ends or game completed, a leaderboard screen will appear showing the total score.
- Activity should be localized, responsive, should have a tutorial, have a fullscreen button.
- The context should be stored using a journal.

NOTE:

- When the timer is **enabled,** The game will end when the timer ends **but** the puzzle will end when the user forms the target shape and Each game has a number of target shapes or puzzles.
- When the timer is **disabled,** The game will end when the user forms the target shape and Each game has only one target shape or puzzle.

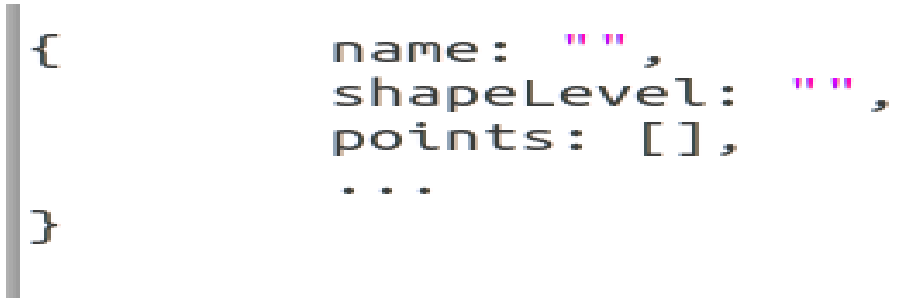
Implementation:
Basic user flow diagram for starting activity:


user starts the activity
present object is null
no
activity starts in MULTIPLAYERS MODE and continue the game using context stored in presence object
yes
datastore object is null
yes
starts new activity instance
no
activity starts with the previous game context and continue the game using context stored in datastore object

- **Generating shape sets**
  We will have various shapes grouped by the shape level i.e. easy, medium, hard.
  For each level, we can have different types of shapes:
  - Letters,
  - Numbers,
  - Birds,
  - Cats,
  - boats etc.
  - others (see here)

  The completed shapes list can be finalized later with mentors in the community bonding period.

  A shape would be defined by a shape object which contains the set of points for the same.



```
{        name: "",
         shapeLevel: "",
         points: [],
         ...
}
```

(the object will contain more fields also)
Using this object we will draw the shape afterwards.

According to the shape level ,which can be selected by the help of a palette in the toolbar, before each game we will generate a random set of shapes for a game.
This will be the set which will be shared with all the users joining the activity.

Now, why are we generating a question set beforehand? Why not generate a question for a game on the fly?

Because, when the user will share the activity, all the users will get the same question set as the host.

- **Enabling / disabling timer**

  If the user enables or disables the timer, the new game starts automatically.

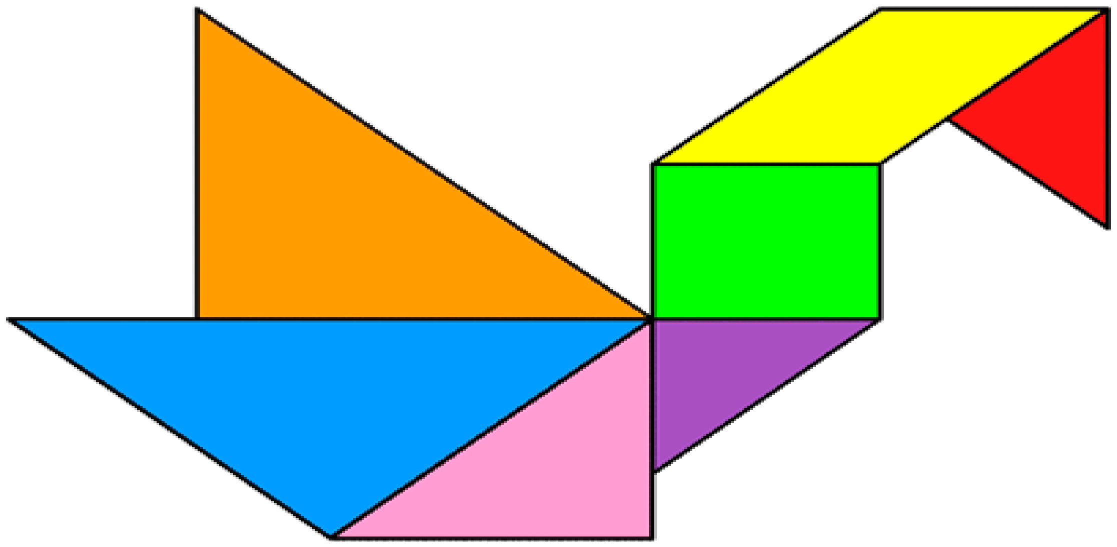
  Multiplayer mode has always timer enabled.

- **Drawing game's target shape**

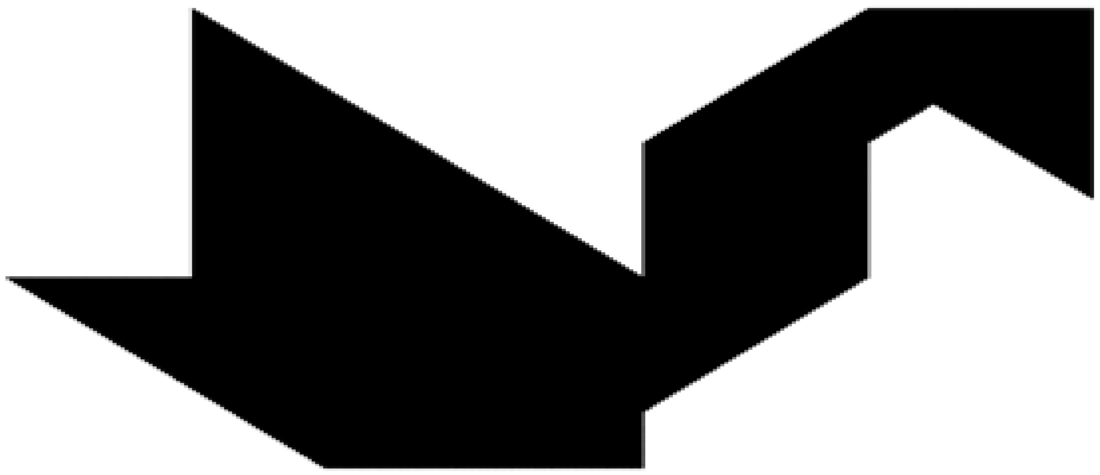
  During the starting the game, we will fetch the shape from the shapes set we have generated before or fetched from the journal. We will then store the points from the shape object into **shapePoints[]**.

  We will then draw the shape using these points.

  Now if the game **difficulty level is easy**, users will know where to put each piece so while drawing the shape we will draw the section and each section will have the same color as its corresponding tangram pieces.



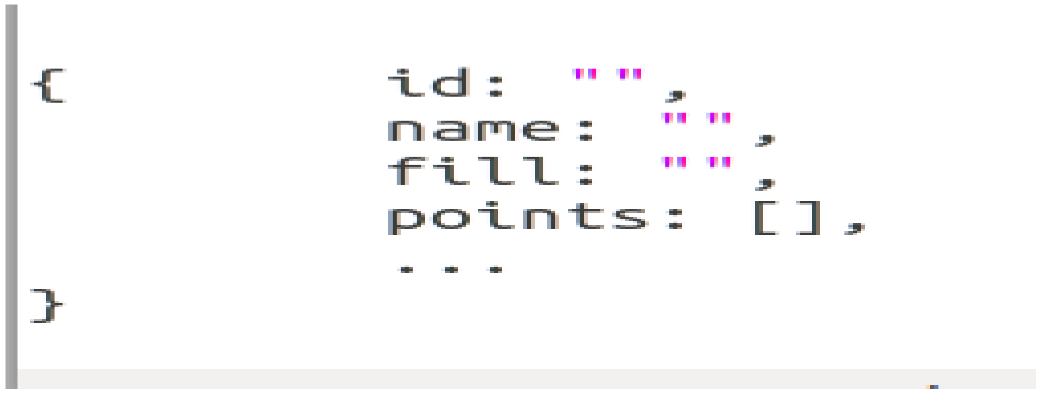
Now if the game **difficulty level is medium** , users have to guess where each piece should be set. So the shape will have no sections like below.



- **Drawing Tangram pieces**

  Tangram has a total of 7 pieces. Each piece will be represented by a piece object which will be predefined in the activity.

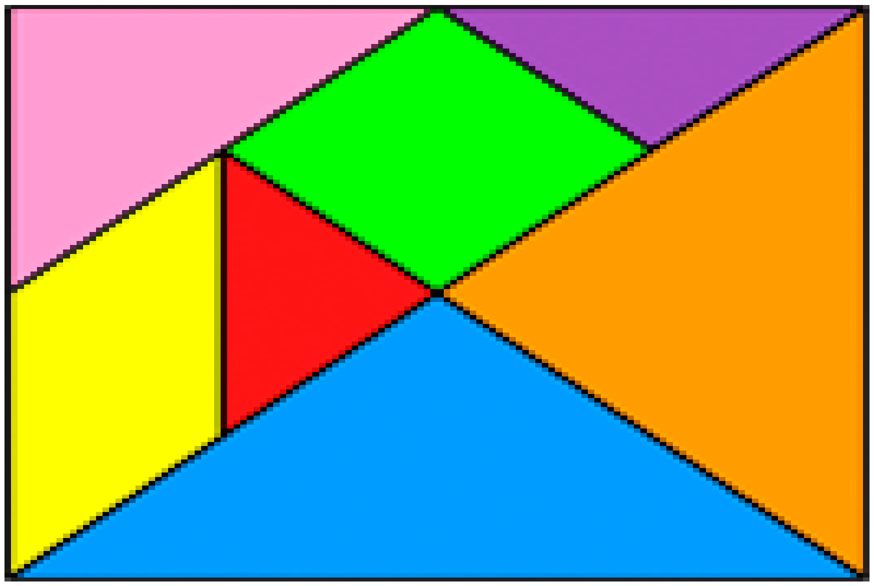
  Piece object would be like:



```
{               id:  "",
                name:  "",
                fill:  "",
                points:  [],
                ...
}
```

  (the object will contain more fields also)
  All the tangram pieces will be of different colors so that users can distinguish between them easily.

  All the tangram pieces will be situated at the **tangram piece section** like below**.**



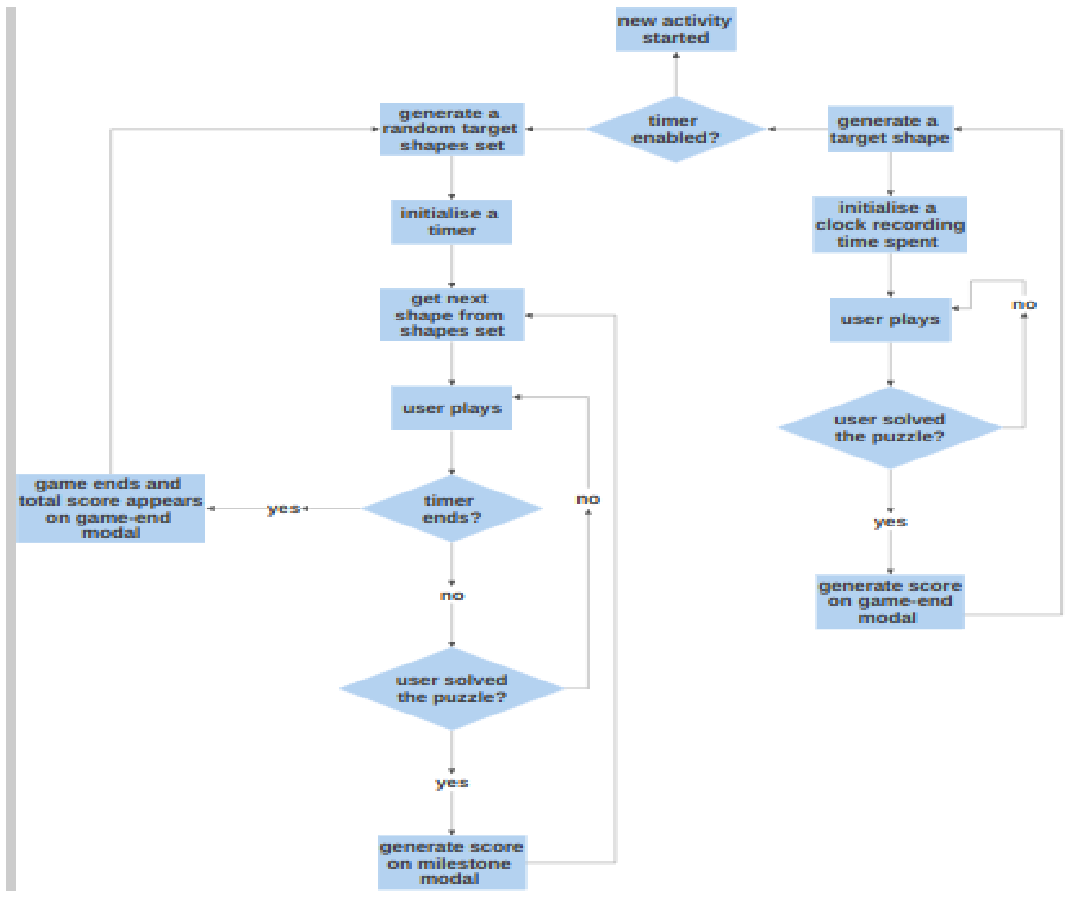
  We will then draw the shape using these points.

- **Playing the activity**

  Users can drag and drop the pieces from one place to another. And then assemble these pieces on the **canvas section**. While assembling the pieces, users can rotate the pieces. In other words Now users will just have to move/rotate the right piece to the right place.

  Ther user will have the option to enable / disable the timer also.

  For single player mode when the user starts the new activity (**both datastore and presence objects are null**):


new activity started
timer enabled?
generate a random target shapes set
initialise a timer
get next shape from shapes set
user plays
timer ends?
yes
no
game ends and total score appears on game-end modal
no
user solved the puzzle?
yes
generate score on milestone modal
generate a target shape
initialise a clock recording time spent
user plays
no
user solved the puzzle?
yes
generate score on game-end modal

- **Calculating Score:**

  By solving each shape or puzzle, the user will get a score for that puzzle. And after the timer ends , the user will get a total score which is basically the summation of all the scores of the puzzle solved in that game.If the user didn’t solve any puzzle fully but partially in a game and the timer ends, the user will still get a score depending upon the progress he made.

Score will depend upon:

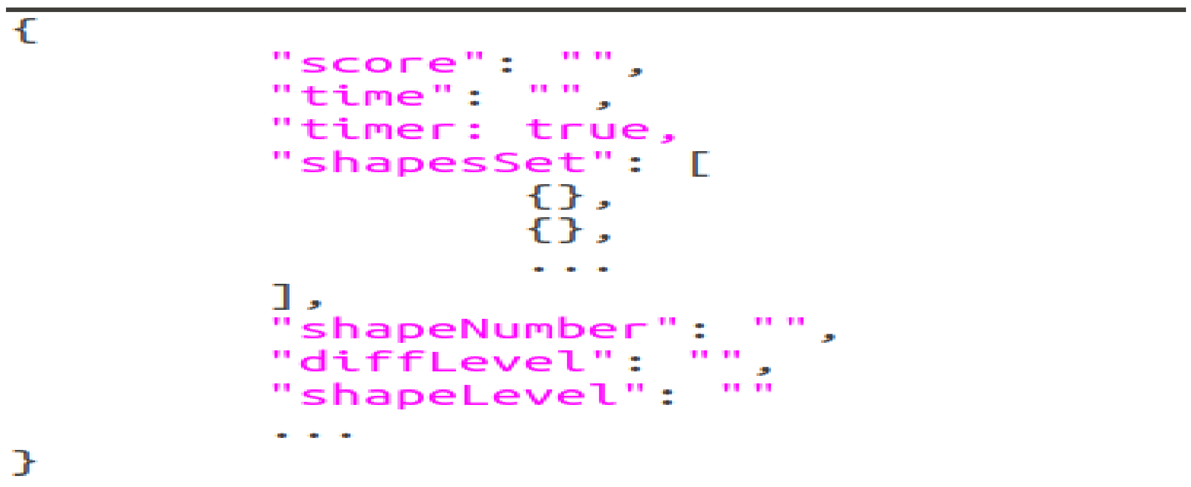
- ○ Number of the pieces set up correctly
- ○ Time spent to complete the puzzle or form the shape

Storing the activity instance:

We will use a journal for storing the activity instance.

The datastore object will look like below:

(The datastore will have some more fields also.)



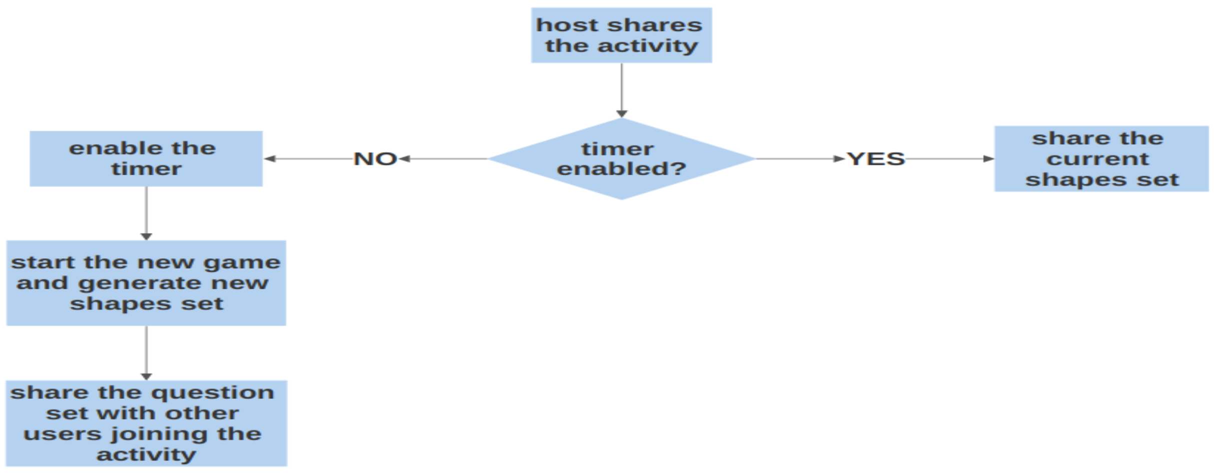
```
{
            "score": "",
            "time": "",
            "timer: true,
            "shapesSet": [
                    {},
                    {},
                    ...
            ],
            "shapeNumber": "",
            "diffLevel": "",
            "shapeLevel": ""
            ...
}
```

Shared activity or multiplayers mode:

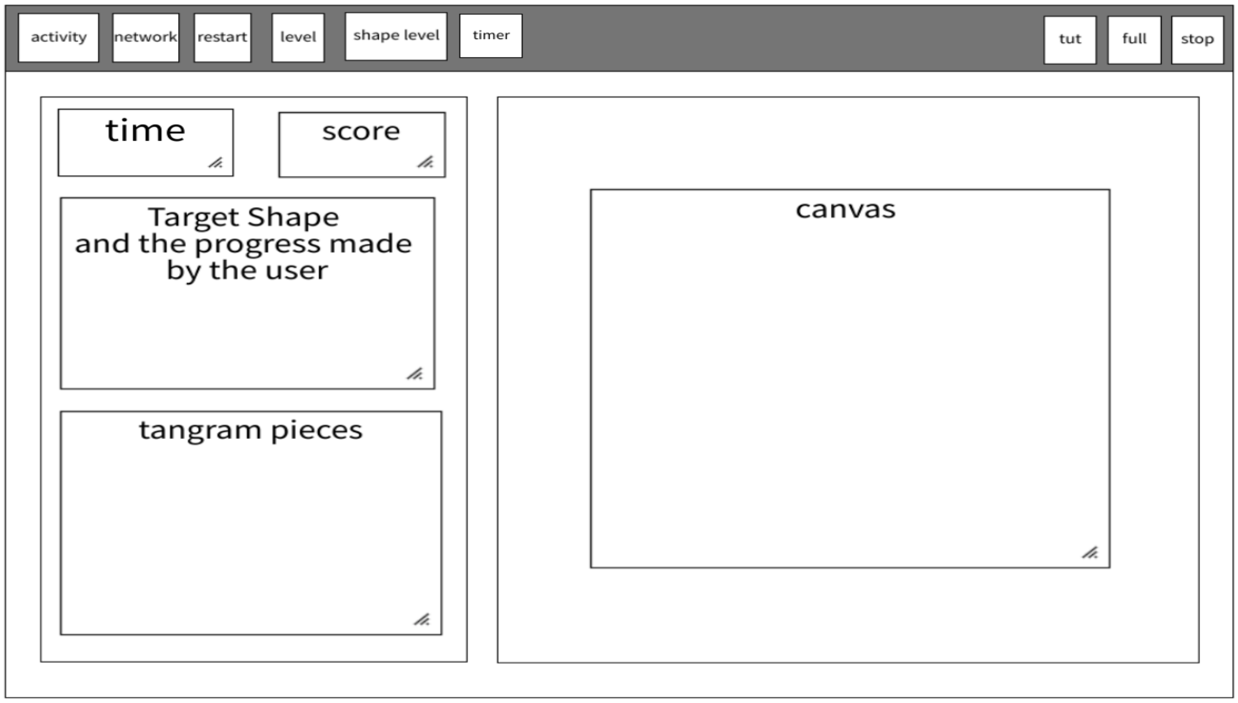
We will use the presence here.
Once the user shares the activity, then users joining the activity via presence will get a shapes set and other info like difficulty level etc.
Once the user ends the game i.e. timer ends, he will get to see the leaderboard.
And the leaderboard will get updated in real-time.

Presence actions :

- **Init:** for initializing the activity with question set,etc.
- **updateLeaderboard:** for updating leaderboard in real time.


host shares the activity
timer enabled?
NO
YES
enable the timer
share the current shapes set
start the new game and generate new shapes set
share the question set with other users joining the activity

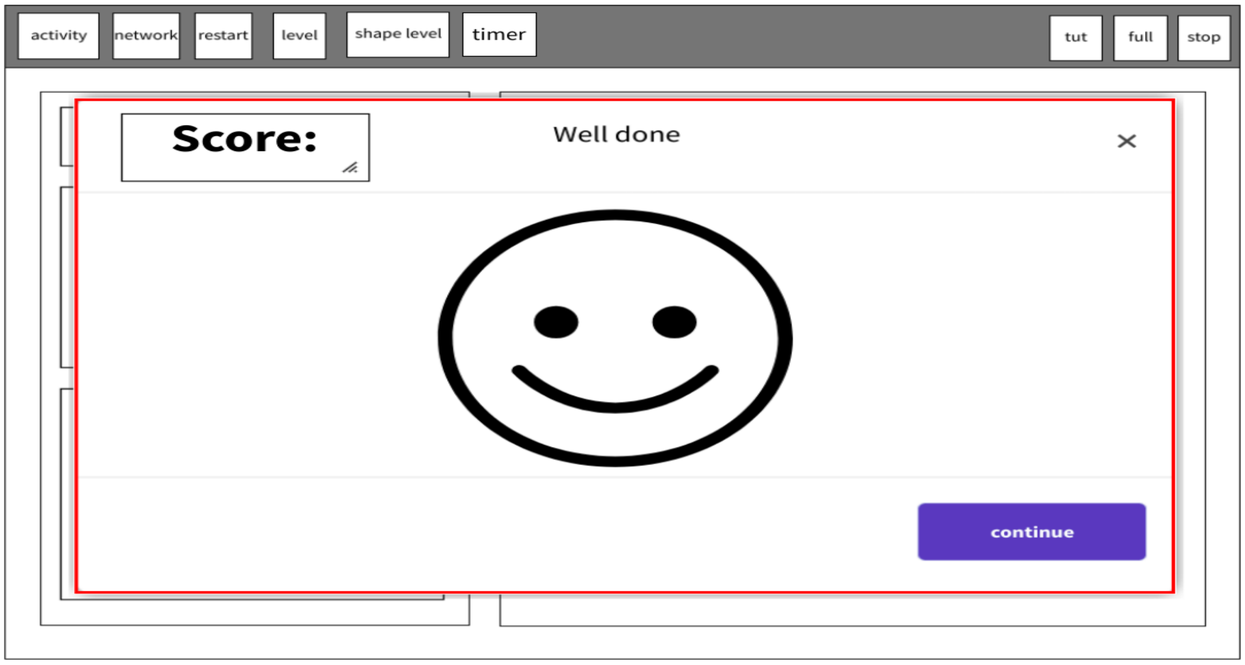
Basic Wireframe for ui:
single player mode:



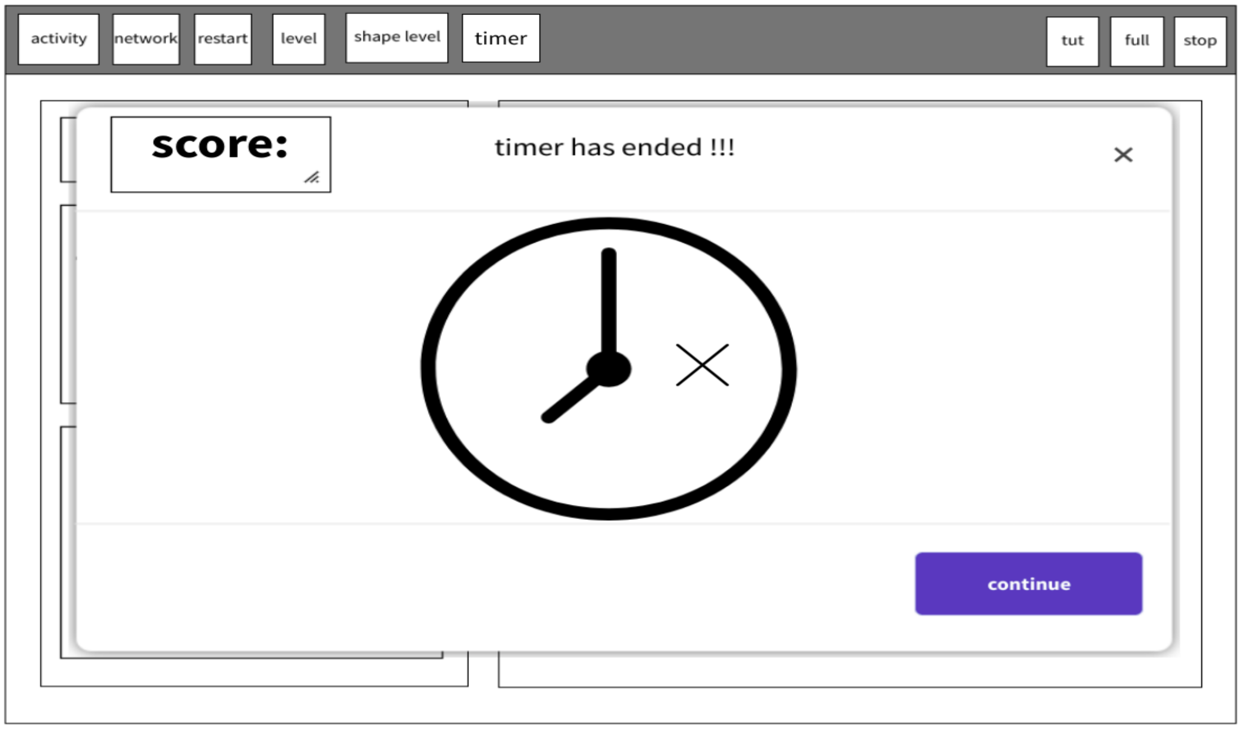
- In single player mode, the screen will have two sections one which will show the tangram pieces, target shape, timer, score etc and another will have a canvas where the user can assemble the pieces.
- If timer is enabled
    - “Time” will show the time left
- If Timer is disabled
    - “Time” will show the time spent so far for the question
- “Score” will show the total score so far.
- Once the user completes the puzzle, a pop up will appear showing its score for the puzzle and the best solution for the same with all steps(see below).

Toolbar in single player mode:

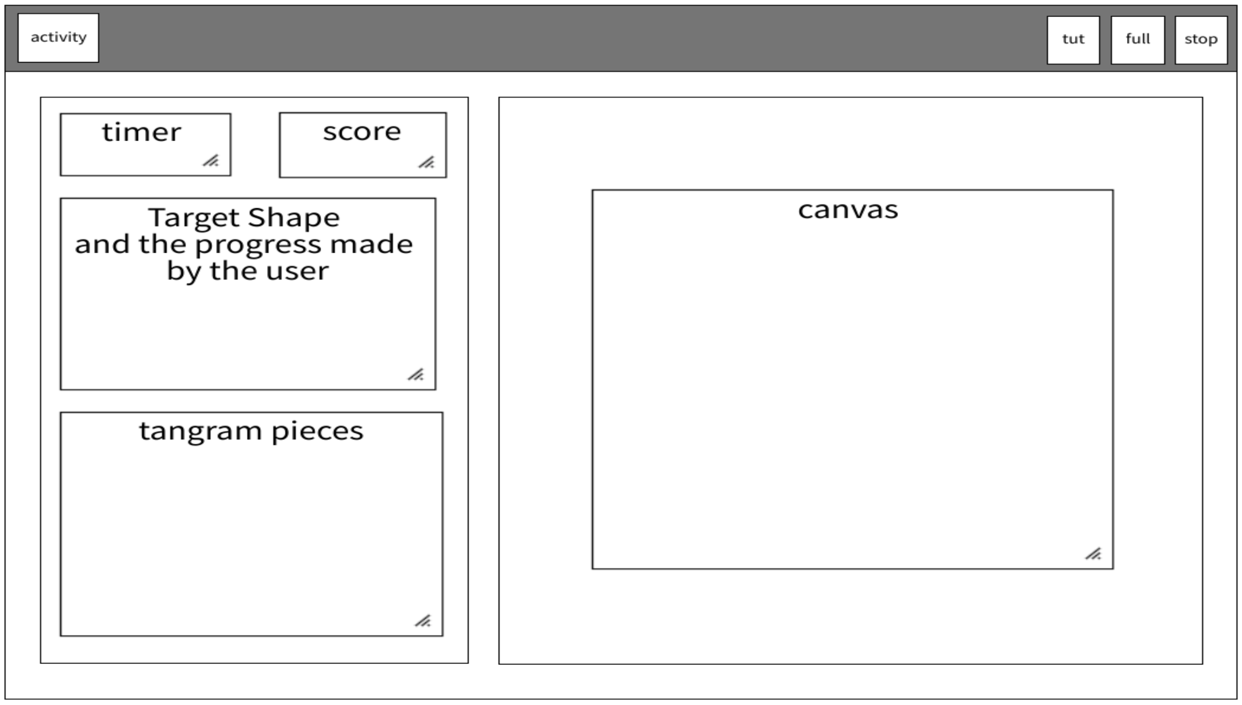
- **Activity:** activity palette
- **Network:** to share activity
- **Restart:** to restart the game with a new question set.
- **Level:** palette to change the difficulty level.
- **Shape Level:** palette to change shape level.
- **Tut:** for tutorial.
- **Full:** for fullscreen/ unfullscreen.
- **Stop:** for stopping the activity or exit.



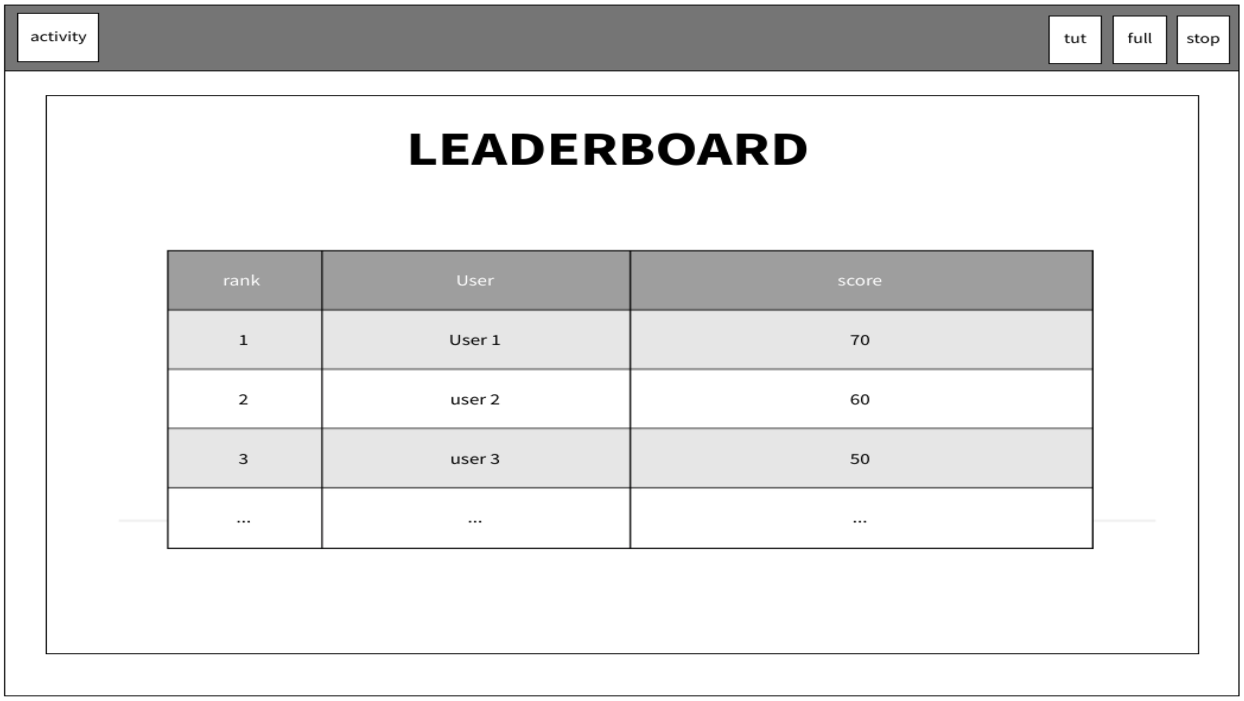
- If **timer is enabled**, then after the game completion or timer ends, the pop up will appear indicating that the time has expired and showing your total score.



Multiplayer mode:



- In multiplayer mode, users will have almost the same ui as that of single player mode except the toolbar.
- After the game completion or timer ends, the leaderboard screen will appear which will show the ranks of users and their points in real time



rank	User	score
1	User 1	70
2	user 2	60
3	user 3	50
...	...	...

Toolbar in multiplayer mode:

- **Activity:** activity palette
- **Tut:** for tutorial.
- **Full:** for fullscreen/ unfullscreen.
- **Stop:** for stopping the activity or exit.

Aim is to build ui as simple as possible.
For framework for ui: we will use vuejs
For building responsive design: we will use bootstrap.

For drawing tangram pieces, shapes and its interaction with user:
To implement this we have following options:

- **Using CSS , jquery ui**
  We can implement tangram game like this
- **Using Fabric.js**

Famous ,powerful and simple javascript html5 canvas library.

- **konva.js**
  Konva is an HTML5 Canvas JavaScript framework that extends the 2d context.
  We can implement tangram game like this
  Konva has provided a vue konva which is a JavaScript library for drawing complex canvas graphics using Vue.

  My decision is to use Konvajs because:
  - Konva js has very good documentation and tutorials.
  - As we are using vuejs and konova provides very good and vuejs wrapper vue konva.

I will take inspiration from following open source implementations to build our game:

- shgalus/tangram
- serkankayaa/konvajs-puzzle
- https://codepen.io/marialilokyee/pen/vNxQpL
- lindsim/tangrams

There are also very helpful links which i will look into to take inspiration and help for building tangram activity

- https://www.gieson.com/Library/projects/games/matter/
- https://www.tangram-channel.

How will it impact sugar labs ?

Both the activities are going to help children to engage themselves in learning new things and their education.
The Mind Math game activity is an engaging **mathematical game** which will encourage students to solve mathematical puzzles and will help in students’ development of computational fluency. Playing mind math game activity will encourage strategic mathematical thinking as students find different strategies for solving problems and deepen their understanding of numbers.

Tangram game activity will be a very joyful activity for children which involves a deep thinking activity which is actually the task of visualizing the spatial relationships between shapes in your "mind's eye" . Research shows that it boosts visual-spatial and arithmetic skills of children.Tangram activity will be a very good activity for sugarizer.

Addition of both the activities to sugarizer (a platform which is dedicated to help children) will promote the more involvement of children with sugarizer.

Mention how much time will you spend each week working on your project

I can easily spend 40-50 hours per week. And can extend my working hours if required.
I will have to take a break of 10 days for my end sems examinations which were previously scheduled on 27 april 2020 but now due to covid-19 pandemic situation, exams can get rescheduled and I don’t have any idea what will be the new schedule as it will depend on how better the situation will be in coming days.
I will compensate for this break in later weeks.

Also discuss your plans after the GSOC period ends.

I will continue to work on this project post the completion of my GSOC tenure as an active contributor and will guide new contributors.
I will keep contributing to sugarizer and will start contributing to sugar python activities to enhance my python development skills and to learn something new.
I have packaged a few sugar applications into flatpak and will keep packaging more applications or activities.
The aim will be the growth and progress of the organization always.

Timeline:

DURATION	TASKS
4 may - 1 june	**community binding period** ● Explore the technologies and libraries that will be used in creating Mind Math and Tangram game activities ● Communicate with my mentors and other org members to finalize the features and UI of the Activity.
1 june - 8 june	● Create a new mind Math activity in Sugarizer and structure the project. ● Start implementing the mind math game activity. ● Start implementing single player mode. ● implement its ui ● build a toolbar. ● Test ● Write a blog
8 june - 15 june	● Study and Implement questions generation algorithms. ● Study and implement tips generation algorithms. ● Integrate the algorithms to the activity ● Add changing difficulty level, and changing compulsory operators features ● Test ● Write a blog
15 june - 22 june	● add a feature to enable / disable timer. ● study and implement a score generation algorithm. ● Add the milestone and timer-end modals or popups ● Add restart feature ● Add Undo feature ● Store and retrieve context from the journal. ● Test ● Write a blog
22 june - 29 june	● Start implementing feature to share activity via presence and implementing multiplayer mode

	• Implement leaderboard screen which will be used only in multiplayers mode. • Improve activity • Test • Write a blog
29 june - 3 july	**phase 1 evaluation** **progress:** • **Single mode of mind math game activity** • **Basic multiplayers mode of mind math game activity**
3 july - 10 july	• Make the required changes after evaluation • Complete the multiplayers mode • Add tutorial • Implement localization via webl10n • Styling of activity • Test activity on various platforms • Try to find and fix bugs • Create a new Tangram game activity in Sugarizer and structure the project • Start implementing Tangram game Activity • Write a blog
10 july - 17 july	• Start implementing single player mode. • implement its ui • Creating tangram's pieces object. • Add drag and drop functionality to the tangram pieces • Add rotate functionality to the tangram pieces • Test • Write a blog
17 july - 27 july	• Add changing difficulty level, and changing shape level features • Add restart feature • add a feature to enable/disable the timer. • Add the milestone and timer-end modals or popups • Start creating different types of target shapes. • Store and retrieve context from the journal. • Test • Write a blog
27 july - 31 july	**phase 2 evaluation**

	Progress: ● **Complete Mind Math game activity** ● **Basic single player mode ui of Tangram game Activity with few target shapes**
31 july - 7 august	● Complete creating all target shapes ● Make the required changes after evaluation. ● Start implementing feature to share activity via presence and implementing multiplayer mode ● Implement leaderboard screen which will be used only in multiplayers mode. ● Test ● Write a blog
7 august - 14 august	● Complete the multiplayers mode ● Implement localization via webl10n ● Add tutorial ● Styling of activity ● Test activity on various platforms ● Test ● Write a blog
14 august - 21 august	● Try to find and fix bugs in both game activities ● Buffer time which can be used if some work left ● Write a blog
21 august - 24 august	● Prepare documentation for both the activities ● Prepare for final evaluation
24 august - 31 august	**final evaluation** ● **Both the activities completed** ● **Documentation of both the activities completed** ● **Both the activities tested thoroughly.**

MISCELLANEOUS

- Google hashcode 2020 global rank: 1855, country rank: 216 certificate
- Set up sugar environment
- Set up sugarizer and sugarizer server environment
- Set up flatpak environment
- Completed GSOC 2020 task here